

SendKeys Statement

See Also [Example](#) [Specifics](#)

Sends one or more keystrokes to the active window as if typed at the keyboard.

Syntax

SendKeys *string* [, *wait*]

The **SendKeys** statement syntax has these named arguments:

Part Description

- string** Required. String expression specifying the keystrokes to send.
- Wait** Optional. Boolean value specifying the wait mode. If **False** (default), control is returned to the procedure immediately after the keys are sent. If **True**, keystrokes must be processed before control is returned to the procedure.

Remarks

Each key is represented by one or more characters. To specify a single keyboard character, use the character itself. For example, to represent the letter A, use "A" for **string**. To represent more than one character, append each additional character to the one preceding it. To represent the letters A, B, and C, use "ABC" for **string**.

The plus sign (+), caret (^), percent sign (%), tilde (~), and parentheses () have special meanings to **SendKeys**. To specify one of these characters, enclose it within braces ({ }). For example, to specify the plus sign, use {+}. Brackets ([]) have no special meaning to **SendKeys**, but you must enclose them in braces. In other applications, brackets do have a special meaning that may be significant when dynamic data exchange (DDE) occurs. To specify brace characters, use {{ } } and { } }.

To specify characters that aren't displayed when you press a key, such as ENTER or TAB, and keys that represent actions rather than characters, use the codes shown below:

Key	Code
BACKSPACE	{BACKSPACE}, {BS}, or {BKSP}
BREAK	{BREAK}
CAPS LOCK	{CAPSLOCK}
DEL or DELETE	{DELETE} or {DEL}
DOWN ARROW	{DOWN}
END	{END}
ENTER	{ENTER} or ~
ESC	{ESC}
HELP	{HELP}
HOME	{HOME}
INS or INSERT	{INSERT} or {INS}
LEFT ARROW	{LEFT}
NUM LOCK	{NUMLOCK}
PAGE DOWN	{PGDN}
PAGE UP	{PGUP}

PRINT SCREEN	{PRTSC}
RIGHT ARROW	{RIGHT}
SCROLL LOCK	{SCROLLLOCK}
TAB	{TAB}
UP ARROW	{UP}
F1	{F1}
F2	{F2}
F3	{F3}
F4	{F4}
F5	{F5}
F6	{F6}
F7	{F7}
F8	{F8}
F9	{F9}
F10	{F10}
F11	{F11}
F12	{F12}
F13	{F13}
F14	{F14}
F15	{F15}
F16	{F16}

To specify keys combined with any combination of the SHIFT, CTRL, and ALT keys, precede the key code with one or more of the following codes:

Key	Code
SHIFT	+
CTRL	^
ALT	%

To specify that any combination of SHIFT, CTRL, and ALT should be held down while several other keys are pressed, enclose the code for those keys in parentheses. For example, to specify to hold down SHIFT while E and C are pressed, use "+(EC)". To specify to hold down SHIFT while E is pressed, followed by C without SHIFT, use "+EC".

To specify repeating keys, use the form {key number}. You must put a space between key and number. For example, {LEFT 42} means press the LEFT ARROW key 42 times; {h 10} means press H 10 times.

Note You can't use **SendKeys** to send keystrokes to an application that is not designed to run in Microsoft Windows

or Macintosh. **Sendkeys** also can't send the PRINT SCREEN key {PRTSC} to any application.

```

End Sub
UserForm1.M2CComm1.OnComm = Transmibuffer
Transmibuffer = "F0AC2" + Chr(13)

UserForm1.M2CComm1.InputMode = comInputModeText
and then set the mode

UserForm1.M2CComm1.OnComm = recvdata
recving data in serial
recvdata = data
UserForm1.M2CComm1.InputMode = comInputModeBinary
now for the data

End If
but I am not sure what to do with it
errormsg return
Else
    it fails
    if successData = "1" then
        successData = UserForm1.M2CComm1.Input
        loop until UserForm1.M2CComm1.InputCount > 1
        DoEvents
    Do
        UserForm1.M2CComm1.InputLen = 1
        rather than use the interrupt for a single byte back
        set to one character when read by input
        UserForm1.M2CComm1.OnComm = Transmibuffer
        Transmibuffer = "FDHEED" + 20(2) + Chr(13)
        D2342 will ignore the space in its conversion
        Transmibuffer = "FDHEED" + 20(range("2e")) + Chr(13)
        Call SetFormOpen
        UserForm1.M2CComm1.InputMode = comInputModeText
        initially we work in text mode

        plyData(range("2e") * 5) + 1) = (msgCheckSum and &HEED0) \ 320 and 322
        plyData(range("2e") * 5) = msgCheckSum and 322
        changing this
    Next If
        maintaining checksum
        msgCheckSum = msgCheckSum + msgData

        plyData(range("2e") * 5) + 1) = msgData \ 320
        msgData1 = msgData \ 320
        msgData1 = msgData and &HEED0
        local significant plus plus
        plyData(range("2e") * 5) = msgData and 322

    End If
    msgData = msgData
    force into an integer
Else
    msgData = msgData * -1
    msgData = msgData

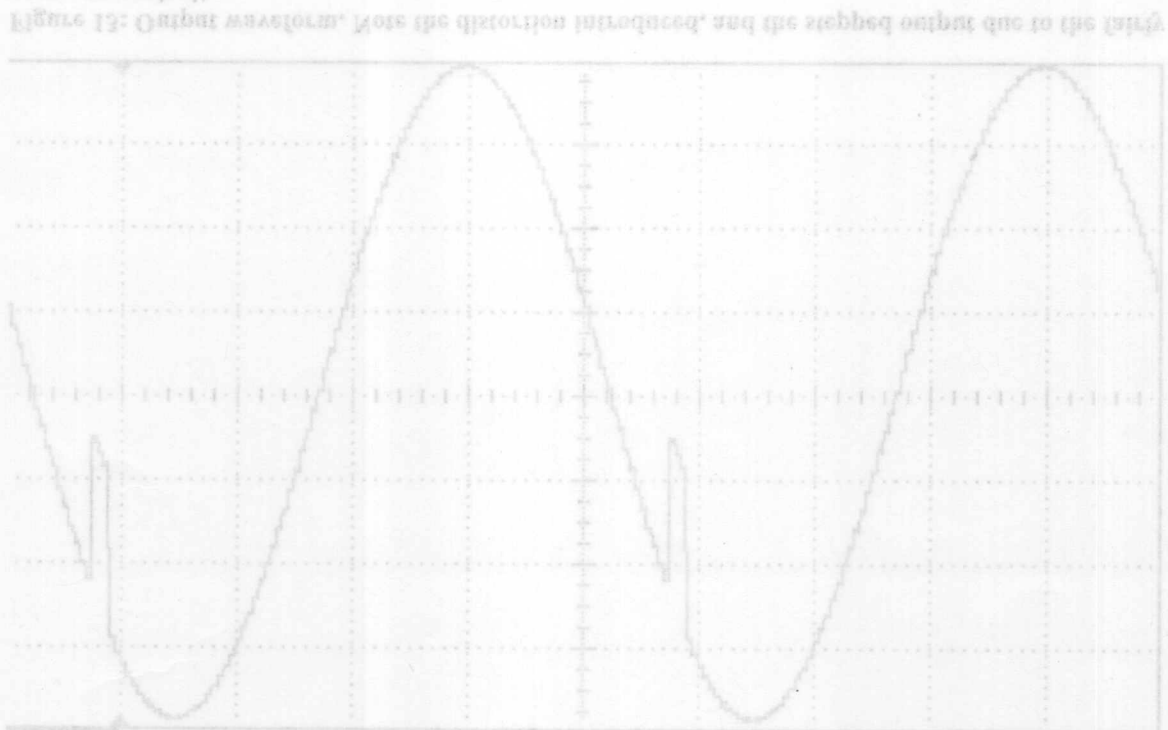
```

SendKeys Statement Example

This example uses the **Shell** function to run the Calculator application included with Microsoft Windows. It uses the **SendKeys** statement to send keystrokes to add some numbers, and then quit the Calculator. (To see the example, paste it into a procedure, then run the procedure. Because **AppActivate** changes the focus to the Calculator application, you can't single step through the code.). On the Macintosh, use a Macintosh application that accepts keyboard input instead of the Windows Calculator.

```
Dim ReturnValue, I
ReturnValue = Shell("CALC.EXE", 1) ' Run Calculator.
AppActivate ReturnValue ' Activate the Calculator.
For I = 1 To 100 ' Set up counting loop.
    SendKeys I & "+", True ' Send keystrokes to Calculator
Next I ' to add each value of I.
SendKeys "=", True ' Get grand total.
SendKeys "%{F4}", True ' Send ALT+F4 to close Calculator.
```

Figure 13: Output waveform.



calls the Download procedure. Figure 13 shows the resulting output on an oscilloscope. Working on either sheet, I placed command buttons on both and the click event simply since the detection of a user induced change could be quite complex. Since the user could I decided that this procedure should only be run from a specific user button command.